Communications and Information EngineeringProgram
Information Theory and Coding (CIE 425)
Fall 2022

Video Compression Project

Construct a team of ***3 members*** and work **together** on implementing your own video encoder/decoder using MATLAB (or Python, C, C++).
You can use the H.264 standard (*from the video compression book on Google classroom*) as a guide in your implementation. You can use any video of your choice and transform it into gray scale level before starting the project.
It is required to perform the following tasks along with their documentation:

A. Video Encoder: [10 Points]:

1) Specify the I and P frames in your video.
2) Implement JPEG encoder of the I-frames. The JPEG encoder should include the following: [2 Points]
   a) Read and divide the image into blocks of 8x8 pixels
   b) Perform DCT on each block (develop your own DCT, don't use the ready-made function in MATLAB)
   c) Perform the quantization step per 8x8 block using a quantization table
   d) Transform each block from 2-D into 1-D vector using the pattern from lecture 5
   e) Use run-length encoding to compress the stream of zeros that may results due to the quantization (use your own developed code)
   f) Use your own developed Huffman encoder to encode the final stream into a further compressed bit stream
3) For the P-frames you need to implement the motion estimation block using a fixed macro-block size. The resolution of the estimator should be one pixel (subpixel estimation is not required in this project). The estimation can be based on one frame (previous I-frame or P-frame) [2 Points]
4) Motion compensation, i.e., creation of the predicted frame. [2 Points]
5) DCT Transforming, quantizing, reordering, run length coding of the estimated residuals from the motion compensation block. The IDCT and rescaling in the encoder has to be implemented as well as *in Fig. 3.50 in the H.264 book*. [2 Points]
6) Entropy coding using your own Huffman function to encode the bitstream + motion vectors [2 Points]

B. Video Decoder: [10 Points]:

1. Implement the JPEG decoder of the I-frames. The JPEG decoder should include the following: [2 Points]
   a. Use your own developed Huffman decoder function to decode the Huffman encoded stream
   b. Perform run-length decoding (use your own developed code)
   c. Transform the 1-D vector into groups of 8x8 matrices
   d. Multiply each group by the quantization tables
   e. Perform IDCT using your own developed function on each 8x8 pixel group
   f. Combine the 8x8 pixel groups into a single image and save it back to a file

2. Entropy decoding using your own Huffman function to extract vectors and headers for each macroblock [2 Points].
3. Reordering, rescaling, IDCT Transforming . The IDCT and rescaling in the decoder has to be implemented as well as *in Fig. 3.51 in the H.264 book*. [2 Points]
4. Motion compensation, i.e., creation of the predicted frame. [2 Points]
5. Reconstruct the macroblocks to produce the decoded frames. [2 Points]

C. Compression Ratio:[4 Marks]:

1. A sample video with and without compression indicating the compression ratio. [2 Points].
2. Compare between the compression ratios for both the low quantization and the high quantization. [2 Points]

D. Evaluation Test: [1 Mark]:

The evaluation test has both objective and subjective evaluation and is used to measure the video Quality. ***mean opinion score*** : the audience would watch the compressed video and see the video quality and score it . It has grades from [5 to 1].

MOS	Quality
5	Excellent
4	Good
3	Fair
2	Poor
1	Bad

Bonus Task: [2 Marks]

Repeat all the steps of the video encoder and decoder with Implementing entropy coding and decoding using your own ***Arithmetic coding function***. Define if there is a difference regarding the compression ratio.

Important Note:

- Documentation of your Project is a mandatory requirement in this part.

  (The project submitted without documentation will not be accepted).

- The documentation is required along with the code itself **either MATLAB Live Editor, or Python Jupyter Notebook then transforming it also to PDF format**
- Documentation should be explained clearly all your used Blocks and their corresponding inputs, outputs, internal variables, etc., and how they map to the implemented task
- The code should be readable and all the variable names are meaningful.
- The code should be commented to explain the functionality of your code.
- The relevant outputs should be displayed in your live editor according to the above tasks.

This is a short video about the MATLAB live editor:
https://www.youtube.com/watch?v=bu4g8ID3aEk

And another one about Python Jupyter Notebook:
https://www.youtube.com/watch?v=3C9E2yPBw7s

Project Logistics:

1. Any plagiarized reports or codes, either fully or partially, will receive **ZERO** points.
   (***This applies to both the original and the copy***)
2. **No late submissions** are allowed.

Grading criteria (Total grade of this project is worth 30% of the course grade)

The **30** points are distributed as follows:

- **5** points: individual discussion at the end of the semester.

  (***The discussion session will be scheduled at a later date***)
- **25** points distributed among the tasks mentioned above (each task has its grade next to it).
  Note that the grade per task includes the documentation of the task.
  The task grading will be on a level from 0-4.
  - **Level 0:** Task not done
  - **Level 1:** Inadequate
  - **Level 2:** Needs improvement
  - **Level 3:** Meets Expectation
  - **Level 4:** Outstanding